PASSED

Group_report

Akshayaswuroupikka Balasubramanian,yixuan xu

March 16, 2016

Assignment 1: Genetic algorithm

In this assignment,we will try to perform one-dimensional maximization with the help of a genetic algorithm.

1.1

The function $f(x) = \frac{x^2}{e^x} - 2e^{\frac{-9sin(x)}{x^2+x+1}}$ is written below:

```
func<-function(x){
     f_x=(x^2/exp(x))-2*exp((-9*sin(x))/(x^2+x+1))
     return(f_x)
}
```

1.2

The function "crossover" that for two scalars x and y returns their "kid" as $(x + y)/2$ is given:

```
cross_over<-function(x, y){
    kid=(x+y)/2
    return(kid)
}
```

1.3

The function "mutate" that for scalar x returns the result of the integer division x^2 *mod* 30 is written:

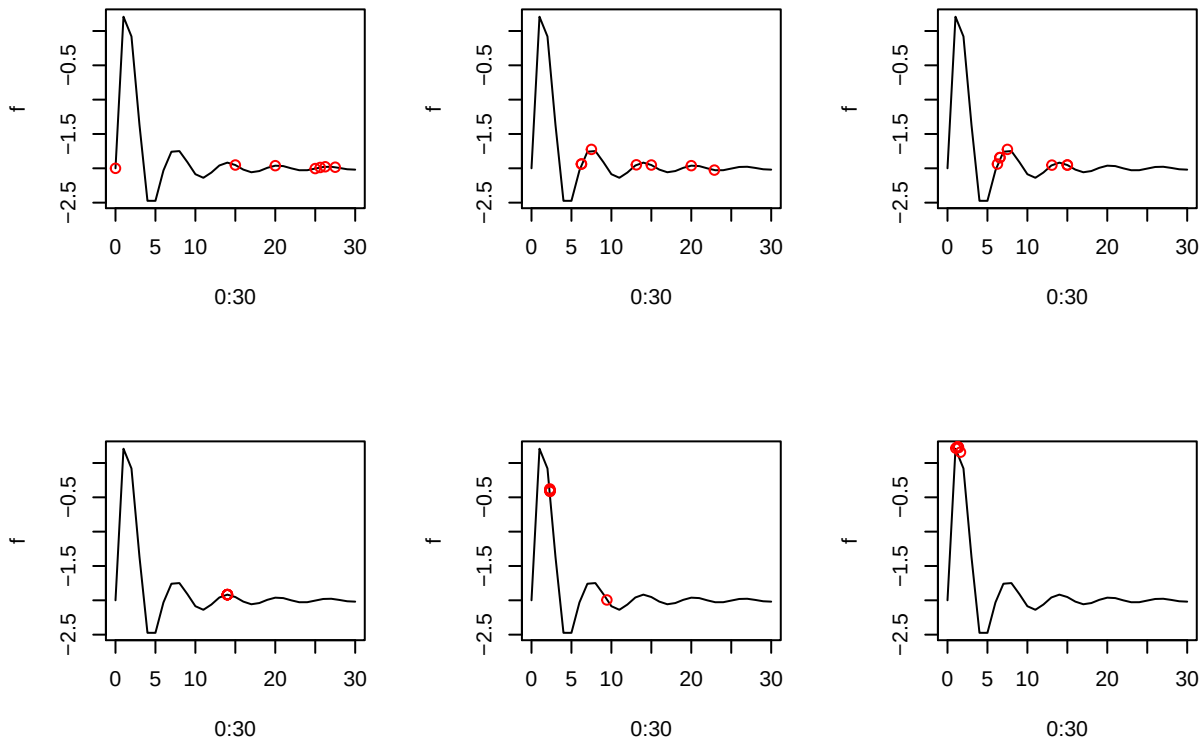
```
mutate<-function(x){
  m=(x^2)%%30
  return(m)
}
```

1.4

A function that depends on the parameters *maxiter* and *mutprob* is done below:

1.5

We are asked to Run your code with different combinations of maxiter=10, 100 and mutprob=0.1, 0.5,0.9.



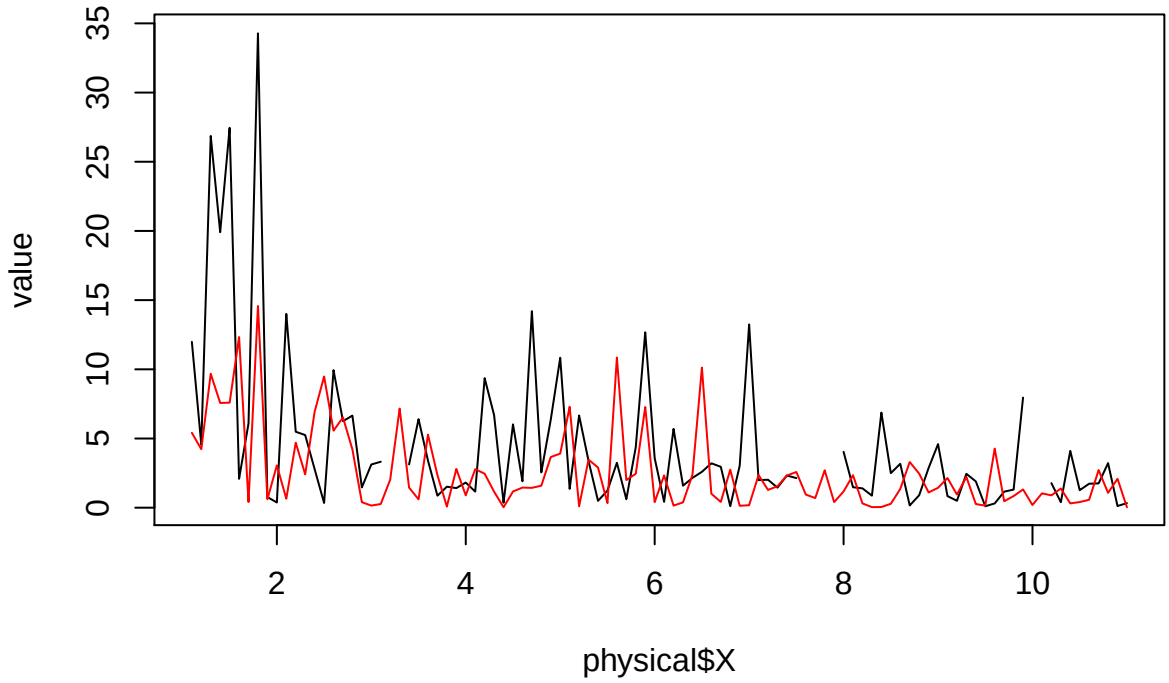
From the plot one could say that if the number of iterations(maxiter) are larger then more points has a more times to move towards global optimum.So if we have minimum number of iterations we get a worst result.But if we have maximum number of iterations then we get almost all points around global optimum.If the probability(mutprob) is 0.5 and above then it has a higher chances of reaching the global optimum.

Assignment 2: EM algorithm

Data file physical.csv describes a behavior of two related physical processes $Y = Y(X)$ and $Z = Z(X)$

2.1

We are asked to make a time series plot describing dependence of Z and Y versus X which is given below:



2.2

Note that there are some missing values of Z in the data which implies problems in estimating models by maximum likelihood. Use the following model

$$Y_i \sim Exp(\frac{X_i}{\lambda}),\ Z_i \sim Exp(\frac{X_i}{2\lambda})$$

where λ is some unknown parameter to derive an EM algorithm that estimates λ.

$$L(\theta) = \prod_{i=1}^{n} \frac{X_i}{\lambda} e^{\frac{X_i Y_i}{\lambda}} \prod_{i=1}^{n} \frac{X_i}{2\lambda} e^{\frac{X_i Z_i}{2\lambda}}$$

$$= \frac{\prod_{i=1}^{n} X_i}{2^n \lambda^{2n}} e^{-\sum_{i=1}^{n} (\frac{X_i Y_i}{\lambda} + \frac{X_i Z_i}{2\lambda})}$$

$$logL(\theta) = 2log(\prod_{i=1}^{n} X_i) - nlog2 - 2nlog\lambda - \sum_{i=1}^{n} \frac{X_i Y_i}{\lambda} - \sum_{i=1}^{n} \frac{X_i Z_i}{2\lambda}$$

Where:

$$\sum_{i=1}^{n} \frac{X_i Z_i}{2\lambda} = \sum_{O} \frac{X_i Z_i}{2\lambda} + \sum_{M} \frac{X_t Z_t}{2\lambda}$$

E-step:

$$E(\frac{1}{2\lambda} \sum_{M} X_t Z_t$$

$$\frac{1}{2\lambda}\sum_{M} X_t E(Z_t) = 2n\lambda_t$$

where $E(Z_t) = \frac{2\lambda_t}{X_t}$

M-step:

$$\frac{\mathrm{d}E(logL(\theta))}{\mathrm{d}x} = \frac{2n}{\lambda} + \frac{1}{\lambda^2}\sum_{i=1}^{n} X_i Y_i + \frac{1}{2\lambda^2}\sum_{O} X_i Z_i + \frac{2|M|\lambda_t}{2\lambda^2} = 0$$

$$\lambda_{t+1} = \frac{2\sum_{i=1}^{n} X_i Y_i + \sum_{O} X_i Z_i + 2|M|\lambda_t}{4n}$$

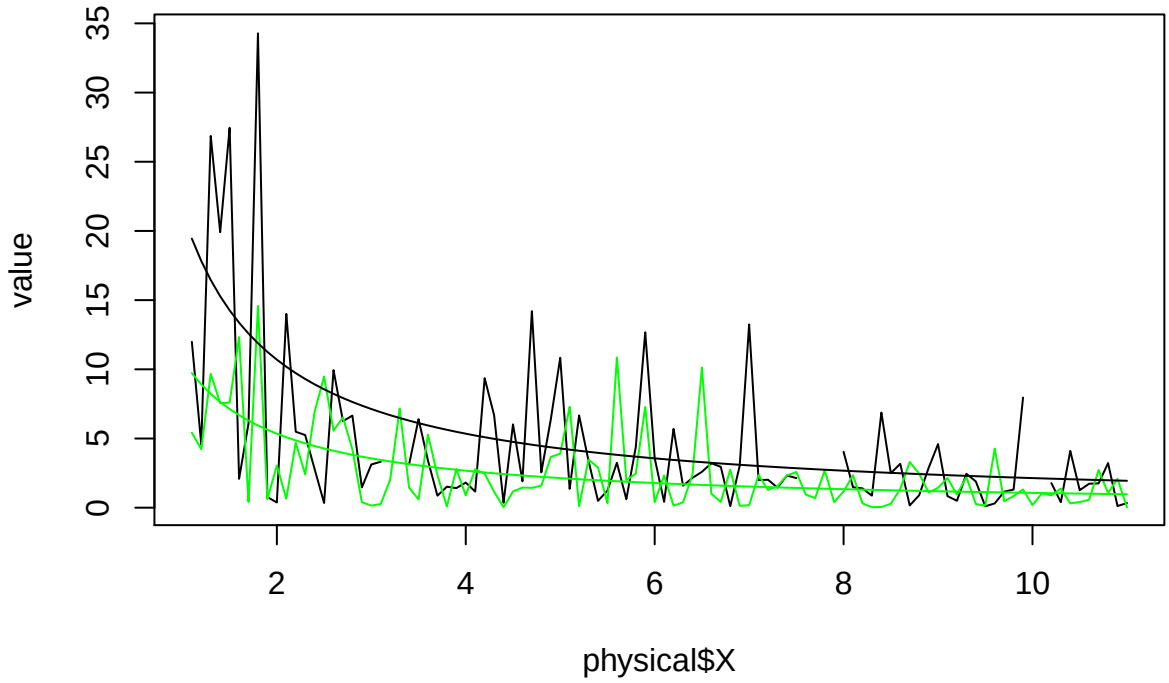
2.4

We have implemented the above given algorithm in R and used $\lambda_0 = 100$ and convergence criterion "stop if the change in λ less than 0.001".

```
## $optimallambda
## [1] 10.69566
##
## $step
## [1] 5
```

2.4

we have plotted EY and EZ versus X in the same plot as Y and Z versus X and stated whether the computed λ seems to be reasonable.



The plot is shown that λ seems to be reasonable, the smooth line has covered the original curve. So lambda is convergence.

conribution

The first part of the assignment is given by akshaya and the second part is given by yixuan xu.

Appendix-R-code

```
## ------------------------------------------------------------------------
func<-function(x){
     f_x=(x^2/exp(x))-2*exp((-9*sin(x))/(x^2+x+1))
     return(f_x)
}

## ------------------------------------------------------------------------
cross_over<-function(x, y){
    kid=(x+y)/2
    return(kid)
}

## ------------------------------------------------------------------------
```

```r
mutate<-function(x){
  m=(x^2)%%30
  return(m)
}

## ----echo=FALSE-----------------------------------------------------------
genetic_func<-function(maxiter,mutprob){
   #Plots function f in the range from 0 to 30
    f=func(0:30)
    plot(y=f,x=0:30,type="l")
   #Defines an initial population for the genetic algorithm as X=(0,5,10,15,.30)
    initial= seq(0,30,5)
    #Computes vector "Values" that contains the function values for each population point
    values=func(initial)
    #Performs maxiter iterations where at each iteration
    maximum_value=0
    for(i in 1:maxiter){
      #Two indexes are randomly sampled from the current population, they are further used as parents (
         parents=sample(initial,size = 2,replace=FALSE)
     #One index with the smallest objective function is selected from the current population, the point
          Order=order(values)[1]
          #victim=initial[Order]
     #Parents are used to produce a new kid by crossover. Mutate this kid with probability mutprob. (us
         new_kid=cross_over(parents[1], parents[2])
         prob=runif(1,0,1)
         if(prob<=mutprob){
           new_kid=mutate(new_kid)
         }
      #The victim is replaced by the kid in the population and the list "Values" is updated.
           initial[Order]=new_kid
           values[Order]=func(initial[Order])
      #The current maximal value of the objective function is saved
           maximum_value[i]<-max(values)


    }

    #Final observations are added to the current plot and marked by some other color.
    points(x=initial,y=values,col="red")

}

## ----echo=FALSE-----------------------------------------------------------
#Run your code with different combinations of maxiter=10, 100 and mutprob=0.1, 0.5,0.9.
par(mfrow=c(2,3))
one=genetic_func(10,0.1)
two=genetic_func(10,0.5)
three=genetic_func(10,0.9)
four=genetic_func(100,0.1)
five=genetic_func(100,0.5)
six=genetic_func(100,0.9)
par(mfrow=c(1,1))
```

```
## ----echo=FALSE----------------------------------------------------------
physical <- read.csv("physical.csv")
plot(y = physical$Z,x = physical$X,type = "l", ylab = "value")
points(y =physical$Y,x = physical$X,type = "l",col = "red")

## ---- echo=FALSE----------------------------------------------------------
eme <- function(data, initlambda, stopcond){
  X <- data$X
  Y <- data$Y
  Z <- data$Z
  Zobs <- Z[!is.na(Z)]
  Zmiss <- Z[is.na(Z)]
  Xobs <- X[!is.na(Z)]
  Yobs <- Y[!is.na(Z)]
  n <- length(c(Zobs, Zmiss))
  r <- length(Zobs)
  #Difine log-likelihood function
  ll <- function(X,Y,Z,lambda,n){
    2*log(prod(X))-n*log(2)-2*n*log(lambda)-as.numeric(t(X)%*%Y)/lambda-
      as.numeric(t(X)%*%Z)/(2*lambda)
  }
  llld1 <- ll(Xobs,Yobs,Zobs,initlambda,n)
  count <- 0
  repeat{
    # E-step:
    newlambda <- (2*sum(X*Y)+sum(Xobs*Zobs)+(n-r)*2*initlambda)/(4*n)
    initlambda <-newlambda
    llld <- ll(Xobs,Yobs,Zobs,initlambda,n)
    count <- count +1
    if(abs(llld1-llld) < stopcond) break
    llld1 <-llld
  }
  return(list(optimallambda = initlambda, step = count))
}
E <- eme(physical,100,0.001)
E

## ----echo=FALSE----------------------------------------------------------
EY <- E$optimallambda/physical$X
EZ <- (E$optimallambda*2)/physical$X
plot(x = physical$X, y = physical$Z, type = "l", ylab = "value")
points(x = physical$X,y=physical$Y,type = "l",col ="green")
points(x = physical$X,y=EY,type = "l", col = "green")
points(x = physical$X,y=EZ, type = "l")

## ----code=readLines(knitr::purl("Group_report.Rmd", documentation = 1)), eval = FALSE----
## NA
```